SGML-dokumenttien automaattisesta muuntamisesta

Heikki Toivonen

Luk-tutkielma
Matematiikan laitos
Jyväskylän yliopisto

17.1.1997

Sisällys

1 Johdanto **2**

2 Yleistä muunnoksista **5**

3 SGML-muunnokset **6**
3.1 Muunnokset yhden DTD:n puitteissa — editointi 7
3.2 Muunnokset DTD:stä toiseen 8
3.2.1 Muunnosten perusteet 8
3.2.2 Dokumenttityyppimääritys puuttuu 9
3.2.3 Muunnosohjelmia 10

4 Tietokantamuunnokset **10**
4.1 Tietokantojen ja tekstidokumenttien eroja 11
4.2 Tutkimusta 12
4.3 Multidoc PRO 14
4.4 Astoria 14

5 Muut muunnokset **15**
5.1 SGML:stä julkaisuksi 15
5.2 Erilaisista formaateista SGML:ksi 16

6 Yhteenveto **16**

Lähteet **18**

A Ohjelmistoja **20**

B Dokumenttityyppimääritykset **20**

1 Johdanto

What you see is all you get. *Brian Kernighan*

Tässä tutkielmassa käsitellään *Standard Generalized Markup Language* (SGML) [ISO86] -standardin mukaisia dokumentteja koskevia automaattisia muunnosohjelmia ja muunnoksiin liittyviä ongelmia. Johdannon jälkeen käsitellään muunnoksia yleensä. Luvussa 3 keskitytään muunnoksiin, joissa sekä lähtö- että maalidokumentit ovat SGML-muotoisia. Luvussa 4 keskitytään tietokantoja koskeviin muunnoksiin. Loput muunnokset käsitellään luvussa 5.

Standard Generalized Markup Language on lyönyt itsensä läpi dokumenttistandardina. Suomessa SGML:ää käyttäviä suuryrityksiä ovat esimerkiksi Nokia, Wärtsilä Diesel ja Valmet. Ulkomaisia tunnettuja SGML:n käyttäjiä ovat mm. USA:n puolustusministeriö ja British Airways. Jyväskylän yliopistokin on rakenteisten dokumenttien tutkimuksen lisäksi ryhtymässä käyttämään SGML:ää esimerkiksi julkaisutietokannan muodossa.

SGML:n historiallista kehitystä voitaisiin lähteä seuraamaan kustantamon toiminnasta. Ennen tietokoneita käsikirjoituksien mukaan liitettiin kirjoitetut ohjeet siitä, kuinka kustantamo halusi painotalon painavan kirjoituksen. Kappaleen viereen voitiin esimerkiksi kirjoittaa: "Tämä kappale 12-pisteen Helvetica Medium Italic -kirjasinlajilla, sisennys 1 tuuma." Tätä merkintätapaa kutsuttiin englanniksi nimellä *markup*, josta nimitys on siirtynyt myös nykyään käytettäviin tietokonesysteemeihin. Merkinnät oli helppo kirjoittaa, koska tavallinen kieli kelpasi.

Tietokoneiden tulon myötä otettiin käyttöön myös uudenlaiset merkintäkielet. Yhteensopivuusongelmia ei juuri ollut vielä tässäkään vaiheessa, sillä käsikirjoitukset toimitettiin edelleen paperille kirjoitettuna, josta ne siirsi painokoneiden ymmärtämään elektroniseen muotoon pienehkö ammattilaisten joukko.

Mikrotietokoneiden yleistymisen myötä kirjailijat alkoivat toimittaa tekstejään elektronisessa muodossa. Silloin syntyivät ensimmäiset yleiset tekstin asettelukielet, esimerkiksi nroff kuusikymmentäluvun lopulla. Myöhemmän vaiheen systeemejä edustaa TEX. Eri käyttöjärjestelmät ja tekstinkäsittelyohjelmat alkoivat pian saada aikaan sekavuutta. nroff-merkatun dokumentin muuttaminen TEXiksi ei ole merkintöjen erilaisuuksien vuoksi helppoa.

Todellinen elektronisen tekstinkäsittelyn kulta-aika koitti WYSIWYG-tekstinkäsittelyohjelmien myötä (WYSIWYG = What You See Is What You Get). Esimerkiksi Microsoft Word kuuluu tähän ohjelmaluokkaan. Hankaluutena on, samoin kuin edellisen kauden ohjelmistoilla, asettelukomentojen sijoittelu tekstin sekaan. WYSIWYG-tekstinkäsittelyohjelmaa käytettäessä merkintöjä ei vain yleensä näe. Muunnosongelma tuli vain entistä suuremmaksi, koska tarjolla on entistä useampia käyttäjiä, käyttöjärjestelmiä ja ohjelmia. Lisäksi WYSIWYGin helppokäyttöisyydellä on tapana aiheuttaa turhaakin ominaisuuksilla leikittelyä. Kustantamo, joka voi ottaa vastaan yleisimmillä ohjelmilla ja käyttöjärjestelmillä luotuja käsikirjoituksia, joutuu hankkimaan tai tekemään muunnosohjelmat kymmeniä eri formaatteja varten. On nimittäin muistettava, että kustantamo ei suinkaan laita painoon kirjailijan tekemää tekstiä sellaisenaan. Mahdollisesti teksti käy läpi useita muutoksia, ensin kustantamon omassa käytössä olevaan monipuoliseen formaattiin ja sitten ehkä vielä painamista varten omaan formaattiin.

American National Standards Institute (ANSI) asetti vuonna 1978 työryhmän muodostamaan tekstinmerkintästandardin, joka olisi yksiselitteinen ja niin monipuolinen, että muita merkintätapoja ei tarvittaisi. Työ siirtyi *International Standards Organisation*ille (ISO), ja 1986 syntyi SGML. Voisi sanoa, että se uudisti tekstidokumenttien ajattelun täysin. SGML:llä kuvataan dokumentin **looginen** rakenne ottamatta mitään kantaa esitystapaan. Kustantamoa ajatellen tämä on ideaalista: kaikki kirjailijat voivat toimittaa kirjan SGML-muotoisena dokumenttina, jolloin kustantamolle jää vapaat kädet kirjan visuaalisessa esityksessä. SGML-muotoisen dokumentin käsittely on tietokoneellakin helppoa sen tiukan rakenteisuuden vuoksi. Kirjailijaakin SGML:n käyttö hyödyttäisi: tutkimusten mukaan tavallisilla tekstinkäsittely- ja julkaisuohjelmilla kirjoittavat käyttävät jopa 30% ajastaan informaation etsimiseen ja toiset 30% tyylien käyttämiseen [Arb95]. Lisäksi teknologia uudistuu täydellisesti parin vuoden välein, jolloin joudutaan vanhan laitteiston ja ohjelmistokannan uusimisen lisäksi muuttamaan myös dokumentit uudelle tekniikalle.

Tekstimuotoisten dokumenttien yleisestä taloudellisesta merkityksestä kertonee se, että joidenkin arvioiden mukaan USA:n bruttokansantuotteesta menee 20% uuden informaation tuottamiseen — tästä 90% on tekstimuotoista dokumentaatiota [Arb95].

SGML ei ole suinkaan ainoa standardi rakenteisille dokumenteille. Se lienee kuitenkin laajimmin tunnettu ja käytössä oleva. Muita standardeja ovat esimerkiksi *Office* (tai *Open*) *Document Architecture* (ODA) [ISO89], jossa loogisen rakenteen lisäksi voidaan ilmaista myös tekstin asettelu, ja *Hypermedia/Time-based Structuring Language* (HyTime) [ISO92], jota voidaan pitää laajennettuna SGML:nä; HyTime soveltuu esimerkiksi multimediaesitysten rakenteiseen kuvailuun. HyTimen SGML:ää edistyneemmät linkit ovat saaneet jotkin ohjelmistojen tekijät lisäämään HyTime-ominaisuuksia SGML-selailuohjelmiinkin [Syn94].

Jonkinlainen SGML-tietämys on tarpeen tämän tutkielman ymmärtämiseksi. Riittävä kuvaus on esimerkiksi artikkelissa *A Gentle Introduction to SGML* [TEI].

Lisäksi olisi hyödyllistä tietää jotakin (relaatio)tietokannoista ja ehkä LaTeXista [Lam94]. Kertauksen vuoksi alla on vielä yksinkertainen SGML:n mukainen dokumenttityyppimääritys ja sen mukainen dokumentti selityksineen:

```
<!-- Tämä on kommentti.
     Tästä alkaa dokumenttityyppimääritys: -->
<!DOCTYPE memo [
<!ELEMENT memo    - - (otsikko? , kappale+)>
<!ELEMENT otsikko - O (#PCDATA)>
<!ELEMENT kappale - O (#PCDATA)>
<!ATTLIST kappale
          tyyli   (normaali | tarkea) tarkea>
]>

<!-- Tästä alkaa dokumentti: -->
<memo>
  <otsikko>
    Tämä on vapaaehtoinen otsikko
  </otsikko>
  <kappale>
    Tämä on normaali kappale. Sisennykset eivät ole
    välttämättömiä, mutta ne auttavat ihmisiä loogisen
    rakenteen erottelussa.
  <kappale tyyli="tarkea">
    Tämä on tärkeä kappale!
</memo>
```

Yo. dokumenttityyppimääritys soveltuu esimerkiksi muistilappuja varten. Muistilapussa on vapaaehtoinen otsikko, jota seuraa yksi tai useampia kappaleita. Kappale-elementillä on attribuutti, jolla voidaan ilmaista kappaleen tärkeys; oletusarvona on normaali-tyylinen kappale. Otsikko ja kappale koostuvat normaalista tekstistä, jonka tarkempaan rakenteeseen tämä dokumenttityyppimääritys ei ota kantaa.

SGML:n mukaisessa dokumentissa dokumentin rakenne kuvaillaan rakennemerkinnöillä (engl. *markup*). Merkinnät tunnistaa tavallisesti hakasulkeista (`< >`), vaikka standardi mahdollistaa myös muiden merkkien määrittelyn. SGML-dokumentti jakautuu loogisiin kokonaisuuksiin, joita voidaan kutsua elementeiksi (engl. *tag*). Usein puhutaan myös alkuelementistä (engl. *start-tag*), elementin sisällöstä (engl. *content*) ja loppuelementistä (engl. *end-tag*). Otsikko-elementin tapauksessa nämä olisivat järjestyksessä "`<otsikko>`", "Tämä on vapaaehtoinen otsikko" ja "`</otsikko>`". Yleisesti on määritelty, että "`</`" tarkoittaa sulkevaa elementtiä eli loppuelementtiä.

Lähteistä voisi sanoa sen verran, että viittauksilla standardeihin on lähinnä se merkitys, että kiinnostunut lukija voi niiden avulla löytää helposti tarkat tiedot mainituista standardeista. Varsinaisesti standardeja ei tässä tutkielmassa käsitellä eikä oteta kantaa niissä esitettyihin väitteisiin tai tutkimustuloksiin.

WWW-lähteistä on pidettävä mielessä se, että varsinkin yksityisten henkilöiden ylläpitämät sivut voivat hävitä tai siirtyä ilman varoitusta lyhyellä aikavälillä. Paras tapa lähteissä mainittujen WWW-dokumenttien uudelleen etsimiseksi on käyttää jotakin hakuohjelmaa laittaen hakusanoiksi dokumentin otsikon ja kirjoittajan.

Kaikki tekstissä mainitut ohjelmistot on koottu liitteeseen A. Liitteessä on mainittu ohjelman tekijät sekä WWW-linkki, josta ohjelman voi hakea tai josta sitä voi tiedustella. Liitteessä B on kerrottu, mistä tekstissä mainitut dokumenttityyppimääritykset voi hakea WWW:n avulla. Vaikka tämän tutkielman painopiste onkin — käytännöllisistä syistä — vapaasti saatavilla olevissa ohjelmistoissa, ei se tarkoita, että kaupallisia ohjelmia olisi vähän tarjolla, päinvastoin.

Joillakin ohjelmistovalmistajilla on myös (mielestäni kiusallinen) tapa julkaista dokumentaatio ainoastaan elektronisessa muodossa, joka seuraa ohjelman mukana. Ohjelmaa omistamatta on lähes mahdotonta tutustua tällaisiin lähteisiin.

Kielenkäytöstä voisi sanoa vielä sen verran, että kaikille asioille ei ehkä ole löytynyt virallista suomenkielistä vastinetta. Välttämättä sellaista ei olekaan. Epäselvissä tapauksissa on pyritty kirjoittamaan englanninkielinen vastine sulkuihin suomennoksen jälkeen.

2 Yleistä muunnoksista

There is no panacea. *Bjarne Stroustrup*

Koska SGML kehitettiin ja sitä käytetään nimenomaan suuria tietomääriä käsitellessä, aiheuttaa se lisävaikeuksia ohjelmia suunnitellessa. Tyypillisesti SGML:ää "vaativa" dokumentointi on kooltaan useita satoja megatavuja, eikä se näin ollen mahdu normaalin tietokoneen muistiin kerralla. Konkreettisia esimerkkejä suurista dokumenteista: yhdessä USAn armeijan laivassa voi olla manuaaleja 25 tonnia ja B-1B-pommikoneesta on miljoona sivua dokumentaatiota [Her94]. Siksi ohjelmat on suunniteltava siten, että ne käsittelevät dokumentteja palasina tai vaihtoehtoisesti pitävät yllä jonkinlaista vähän muistia vaativaa tietorakennetta ja hakevat sen avulla muistiin dokumentin osia tarpeen mukaan. Edellinen ratkaisu voi hidastaa paljonkin dokumentin käsittelyä, mutta hyvät välimuistiratkaisut parantavat tilannetta jonkin verran [Syn96].

Erilaisille muunnosohjelmille on yhteistä se, että mitä yleiskäyttöisempiä ne ovat, sitä enemmän käyttäjä joutuu tekemään ohjaustiedostoja tai antamaan muita ohjauskomentoja sovellukselle. Käyttöliittymät eivät ole useinkaan kovin kehittyneitä johtuen ehkä siitä, että useimmat verkosta löytyvät ohjelmat ovat pikaisesti tehtyjä, usein vielä UNIX-ympäristössä komentoriviltä käytettäväksi (tästä hieman lisää kappaleessa 5.2).

Muunnoksia ajatellessa on muistettava, että edelleen suurin osa tietokoneilla olevasta tekstidokumentaatiosta ei ole SGML:ää tai edes rakenteellisessa muodossa. WWW:ltä löytyy mm. maininta projektista, jossa muunnettiin 200 miljoonaa dokumenttia SGML:ksi [Fre].

3 SGML-muunnokset

To iterate is human, to recurse divine. *L. Peter Deutsch*

Tässä kappaleessa keskitytään kuvailemaan muunnoksia, joissa sekä lähtö- että maalidokumentit ovat SGML-dokumentteja. Koska *HyperText Markup Language* (HTML) (katso liite B) -dokumentit voidaan kuvata SGML-standardin mukaisella dokumenttityyppimäärityksellä (*Document Type Definition*, DTD), käsitellään siksi myös HTML-dokumenttien muunnoksia tässä yhteydessä. HTML-lyhenteellä on kaksi merkitystä: HTML DTD ja HTML DTD:n mukaisen dokumentin rakennemerkinnät. On kuitenkin huomattava, että on olemassa useita dokumenttityyppimäärityksiä, joita kaikkia yleisesti vain sanotaan HTML:ksi.

Dokumenttityyppimääritykset, jotka kuvailevat eri tyyppiset SGML-dokumentit, ovat sekä SGML:n voima että heikkous. SGML-standardi on pieni (standardien mittakaavassa) [ISO86], mutta silti erittäin mukautuva ja käyttökelpoinen. Tämä on pitkälti juuri tiettyjä käyttökohteita varten räätälöityjen dokumenttityyppimääritysten ansiota. Heikkous DTD:t ovat sen vuoksi, että ei ole olemassa (eikä voikaan olla) mitään yleispätevää reseptiä, jolla yhtä DTD:tä vastaava dokumentti voitaisiin muuntaa vastaamaan toista DTD:tä.

SGML-dokumenttien muuntamiseen liittyy lähes väistämättä rekursiivisuus. Rekursiivisuus jo sinällään on yksi paljon ongelmia aiheuttava alue ohjelmoinnissa. Dokumentit pitää myös jäsentää (engl. *parse*). Tämä on kuitenkin suhteellisen helppoa rakenteisuuden vuoksi. Ongelmia aiheuttavat edelleen useat DTD:t eli jäsennin (engl. *parser*) pitää tehdä vain tiettyä DTD:tä varten tai sitten sen on pystyttävä jäsentämään myös DTD ja sen jälkeen dokumentti (katso kuitenkin 3.2.2).

SGML-dokumenttien muuntamiset voidaan jakaa kahteen ryhmään: dokumentin muuttaminen vanhan DTD:n puitteissa ja dokumentin muuttaminen vastaamaan uutta DTD:tä.

3.1 Muunnokset yhden DTD:n puitteissa — editointi

Muunnettaessa dokumenttia saman DTD:n puitteissa on kyseessä editointi. SGML-dokumenttien kohdalla kyse on kuitenkin huomattavasti hankalammasta operaatiosta kuin ei-rakenteisen dokumentin kohdalla. SGML-editorin on huolehdittava siitä, että dokumentti vastaa DTD:tä myös editoinnin jälkeen.

Yleinen editorien periaate on muodostaa dokumenttia luotaessa DTD:tä vastaava minimidokumentti, johon käyttäjä täyttää otsikot yms. Lisättäessä jotakin editori esittää valikoilla, mitkä elementit ovat sallittuja lisäyksiä juuri siinä kohtaa. Siis dokumentti on aluksi DTD:n mukainen, ja kun kaikki tehtävät muokkaukset ovat DTD:n mukaisia, pysyy dokumenttikin aina DTD:n mukaisena. Tämä on siis yleisperiaate, mutta siitä ei voida aina pitää kiinni.

Editointiongelmat voidaan jakaa kahteen ryhmään Colen ja Brownin mukaan [Col92]:

- sisäiset rakenneongelmat
  - dokumentin oikeellisuus luotaessa ja editoidessa dokumenttia
  - toiminta vajaiden ja väliaikaisesti ei-DTD:n mukaisten dokumenttien kanssa
  - sallittujen rakenteisten muutoksien tunnistaminen
- käyttöliittymäongelmat
  - vajavaisten dokumenttien näyttäminen
  - käyttäjän opastaminen

Colen ja Brownin mukaan sisäiset rakenneongelmat voidaan suurelta osin hoitaa tunnettujen jäsennintekniikoiden avulla, mutta käyttöliittymään liittyviin ongelmiin he keskittyvät enemmän. Heidän ehdotuksensa on käyttää nk. *fall-back* -luokkia, joilla vähennettäisiin tilanteita, joissa editori tylysti sanoo jonkin operaation olevan DTD:n vastainen. Otetaanpa esimerkki:

Dokumentissa on yksinkertaisia kappaleita, joissa voi olla pelkkää tekstiä, sekä edellisiä monimutkaisempia kappaleita, joissa voi olla alaviitteitä jne. Jos käyttäjä haluaisi kopioida monimutkaisia kappaleita alueelle, joihin käyvät vain yksinkertaiset kappaleet (esimerkiksi dokumentin abstrakti), voidaan määritellä *fall-back* -luokka siten, että ohjelma osaa korvata alaviitteen em. siirrossa vaikkapa kirjoittamalla alaviitetekstin sulkuihin. Käyttäjä olisi oikeutetusti hyvin närkästynyt, jos ohjelma vain huomauttaisi, että kopiointioperaatio on laiton eikä sitä voida suorittaa.

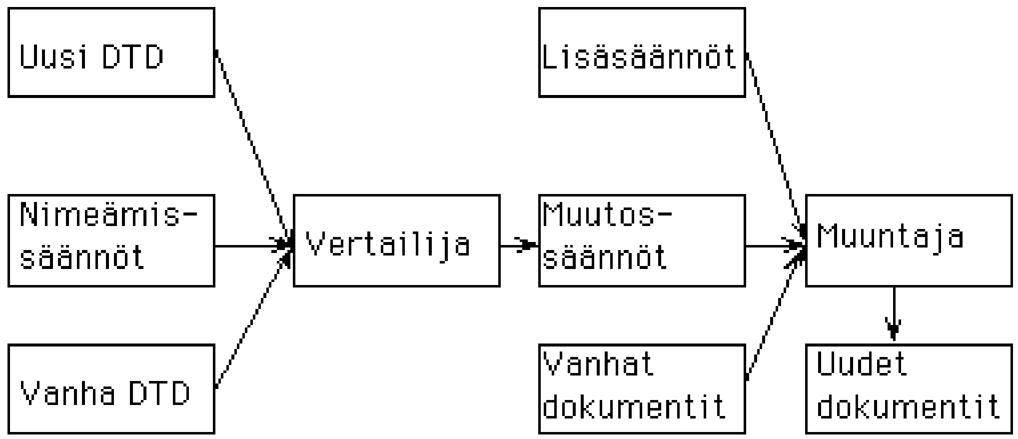
Rakenteisten dokumenttien editoreihin liittyvät ongelmat kuitenkin vain sivuavat varsinaista aihetta, joten ei niistä tämän enempää.

3.2 Muunnokset DTD:stä toiseen

Nykyisin yleisin SGML-dokumenttien muunnosongelma lienee HTML:ksi muuntaminen *World Wide Web*in (WWW) suosion vuoksi; kaikki yleinen dokumentointi tulisi saada WWW:lle. Yleistymässä ovat myös yritysten ja yhteisöjen ns. intranetit (intranetti = yhteisön sisäinen "internet"). Intranetissä myös organisaation sisäinen informaatio voidaan esittää HTML-muodossa. Painetta muunnoksiin siis on.

3.2.1 Muunnosten perusteet

Muunnokset DTD:stä toiseen tuntuvat ensin alkuun lähes mahdottomalta ajatukselta. Akpotsui ja Quint ovat esittäneet kuitenkin yksinkertaisen mallin siitä, kuinka se voidaan (periaatteessa) tehdä [Akp92]; malli on kuvassa 1.



Kuva 1: Muunnosten periaate

Vertailija tekee muutossäännöt kaikille elementtipareille vanhan ja uuden DTD:n välillä. Nimeämissäännöt auttavat parien luomisessa. Nimeämissäännöt tulisi käyttäjän kirjoittaa, sillä niin älykästä tietokonetta ei ole vielä kehitetty, että se pystyisi tässä ihmisen päihittämään. Apuna toki voidaan käyttää joitakin heuristisia periaatteita eli kuvaan voitaisiin lisätä esivertailija, joka antaisi käyttäjälle pohjatiedoston nimeämisiä varten. Seuraavassa listassa on muutamia tilanteeseen sopivia heuristisia periaatteita:

- elementin nimien samankaltaisuus, esimerkiksi vanhassa DTD:ssä esiintyy PAR ja uudessa DTD:ssä PARA
- elementin attribuutin nimien, sallittujen arvojen ja lukumäärän samankaltaisuus
- erinimiset elementit sisältävät samannimiset lapsielementit, esimerkiksi vanhassa DTD:ssä on määritelty `A (PARA , FOOT)` ja uudessa DTD:ssä `B (PARA , FOOT)`
- sanastoa käyttävä ohjelma voisi lisäksi määritellä (lähes) samaa tarkoittavia sanoja yhdeksi elementiksi, esimerkiksi jos vanhassa DTD:ssä esiintyvät elementit MATH ja FORMULA, ja uudessa DTD:ssä MATHEMATICS, voisi esivertailija tarjota muunnossäännöiksi MATH- ja FORMULA-elementtien muuntamisen MATHEMATICS elementiksi

Näistä yleisistä ohjeista huolimatta useimmat käytännön muunnosohjelmistot keskittyvät tiettyjä DTD:tä noudattaviin dokumentteihin. Se ei ole kyllä mikään ihme, kun tutkii tarkemmin millaisia tilanteita muunnoksien kanssa tulee. Erikoistuminen vaikuttaa myös automaatioasteeseen (ks. 3.2.3).

Kuikka ja Penttonen mainitsevat tutkimuksien osoittaneen, että eniten muunnokset koskevat elementtien liikuttelua, uusien elementtien lisäämisiä ja vanhojen tuhoamisia sekä elementin ominaisuuksien muuntamista (eli SGML:n tapauksessa attribuuttien muuntamisia) [Kui93]. Myöskään monimutkaiset tapaukset, kuten elementtien uudelleen ryhmittelyt, eivät ole harvinaisia. Muunnoksissa vaikeuksia lisäävä seikka on se, että muunnosten järjestyksellä voi olla väliä [Toi95].

Joskus lisäongelmana voi olla myös se, että DTD puuttuu.

3.2.2 Dokumenttityyppimääritys puuttuu

Tilanne, jossa joko uusi tai vanha DTD (tai molemmat) on hävinnyt, ei ole niin katastrofaalinen kuin voisi kuvitella. Nykyään on jo ohjelmia, jotka osaavat muodostaa DTD:n SGML-dokumenteista. Tällainen löytyy esimerkiksi Fred-projektilta.

Vaikka dokumentista pystytäänkin saamaan selville sen määrittelevä DTD, se ei välttämättä kuitenkaan ole täydellinen. Saatu DTD kuvaa tarkkaan lähteenä käytetyn dokumentin, mutta se ei ehkä ole kuitenkaan täysin oikein seuraavaa dokumenttia ajatellen. Jos esimerkiksi alkuperäisessä, kadonneessa DTD:ssä on vapaaehtoisia elementtejä tai elementtejä, jotka voivat esiintyä monissa paikoissa, tarvitsisi DTD:tä rakentavan ohjelman käydä läpi kaikki mahdolliset dokumentit, jotka vastaavat kyseistä DTD:tä. Tämä ei selvästikään ole aina mahdollista.

Kun korvaava DTD on saatu riittävän hyväksi (eli se käytännössä pätee kaikille käyttäjän tuntemille dokumenteille), palautuu muunnosongelma taas normaaliksi.

3.2.3 Muunnosohjelmia

Aiemmin mainittiin, että yleisiä muunnosohjelmia on vähän. Erikoistuneita ohjelmia sen sijaan on verkosta helppo löytää. Erikoistuneiden ohjelmien hyvä piirre on siinä, että niistä voidaan tehdä jopa täysin automaattisia, eikä käyttäjän tarvitse määritellä mitään asetuksia tai ohjelmoida lisäyksiä. Eräs erikoistunut sovellus on Linuxdoc-SGML. Linuxdoc-SGML muuntaa QWERTZ DTD:n mukaiset dokumentit HTML:ksi (myös muut formaatit ovat mahdollisia). Systeemi ei ole keskittynyt pelkästään Linuxiin, vaan sillä voidaan muuntaa mitä tekstiä tahansa, kunhan se vastaa QWERTZ DTD:tä, joka taas on kehitetty ohjelmistojen dokumentointiin.

Esimerkki yleisestä muunnosohjelmistosta on Sgmlspl. Se osaa tulkita dokumenttityyppimääritykset ja pystyy muuntamaan SGML-tekstin mihin formaattiin hyvänsä. Sgmlspl käyttää määrittelytiedostoa, jossa elementeittäin määritellään tehtävät toimenpiteet. Hankaluutena tässä on se, että määrittelytiedosto on tehtävä Perl5-kielellä, joka on kaukana käyttäjäystävällisyydestä. Ohjelman mukana seuraavat määrittelytiedostot LaTeX- ja HTML-muunnoksia varten. Esimerkki oman DTD:n mukaisesta muunnoksesta on Kunnarin, Lyytikäisen ja Pietisen pro gradu -tutkielmassa [Kun95].

Monet verkosta löytyvät muunnosohjelmat käyttävät James Clarkin SGMLS-jäsentäjää, joka on vapaassa levityksessä. Jäsentäjässä olevat rajoitukset siirtyvät sellaisenaan sitä käyttäviin sovelluksiin, joten monet muunnosohjelmista ovat samalla tavalla rajoittuneita. Enemmän jäsentäjistä ja SGMLS:stä löytyy lähteestä [Kun95].

4 Tietokantamuunnokset

> Jokaiseen sivilisaatioon sisältyy väistämättä sellainen tiedostamaton voima, joka saattaa estää, kavaltaa tai tehdä tyhjäksi melkein minkä hyvänsä yhteisön tiedostetusti aikoman hankkeen.

Frank Herbert, Dyynin Messias

Tietokantojen esittäminen SGML-dokumenttina, tai ylipäätään minkäänlaisena tekstidokumenttina, tuntuu aluksi hieman oudolta. Tietokannoissa ei ole perinteisesti juuri talletettu kokonaisia dokumentteja, mutta esimerkiksi jonkin voimalan varaosaluettelo voisi löytyä tietokannasta. Kun tähän yhdistetään käyttö- ja huolto-ohjekirjat, alkaa ajatus tuntua mielekkäältä. Olisipa hienoa saada tietokoneelta rikki menneen osan tarkka huolto-ohje ja varaosatiedot omiin tarpeisiin räätälöitynä!

4.1 Tietokantojen ja tekstidokumenttien eroja

Tietokannat ovat olleet käytössä jo pitkään verrattuna rakenteisiin dokumentteihin. Niiden tärkeimpiä ominaisuuksia ovat tehokkaat haut. Tiedon etsiminen on oleellinen osa myös tavallisilla dokumenteilla. SGML-muotoisessa dokumentissa voidaan tehdä paljon monipuolisempia hakuja kuin tavallisessa dokumentissa. Voidaan esimerkiksi hakea dokumentteja, joissa on mainittu kirjailijan osoite (olettaen, että se on DTD:ssä määritelty). Em. haku pelkän tekstin perusteella ei olisi kovin mielekästä, sillä etsimällä esimerkiksi dokumentteja, joissa on sana "osoite", ei löydetä sellaisia dokumentteja, joissa kirjailijan osoite mainitaan ilman "osoite"-sanaa. Vastaavasti voisi löytyä myös dokumentteja, joissa puhutaan osoitteista, mutta ei mainita kirjailijan osoitetta. Asiaa voidaan vielä vaikeuttaa yhdistämällä hakuun vaikkapa puhelinnumero ja se, että dokumenttien tulisi olla tietyn kielisiä. Nämä tiedot on luonnollisesti oltava myös DTD:ssä määriteltynä, jotta niitä voitaisiin käyttää haussa.

Tietokannoissa on yleensä kyse siitä, että tieto voi muuttua nopeassakin tahdissa, mutta tiedon esitystapa tai tietokannan rakenne on hyvinkin pysyvä. Esimerkiksi pankin asiakastietokanta on tällainen: asiakkaan tilitiedot voivat muuttua useita kertoja päivässä, mutta tietokannan rakenne — asiakkaan kuvaileva taulu — tuskin muuttuu edes useassa vuodessa.

Tekstimuotoisen tiedon perinteistä käsittelyä kuvaa se, että kun teksti on kerran kirjoitettu, siihen ei juuri tule muutoksia. Tekstin esittämistapaa sen sijaan on jouduttu muuttamaan siirryttäessä tekstinkäsittelyohjelmasta toiseen tms. Esimerkki tällaisesta tilanteesta on vaikkapa väitöskirja: kerran kirjoitettua väitöskirjaa tuskin kukaan rupeaa muuttelemaan, mutta se voidaan ehkä haluta paperitulostuksena, WWW:llä, CD-ROMilla ja niin edelleen.

Tietokantojen ja dokumentoinnin eroista on erityisesti lähteessä [Ray96].

Näin erilaisista lähtökohdista tulevien medioiden yhdistäminen on ensinäkemältä vaikea tehtävä ja kieltämättä käytännössäkin vaikeuksia on. Yksinkertaisia periaatteita on kuitenkin esitetty ja ohjelmistoja on luotu näiden periaatteiden mukaan.

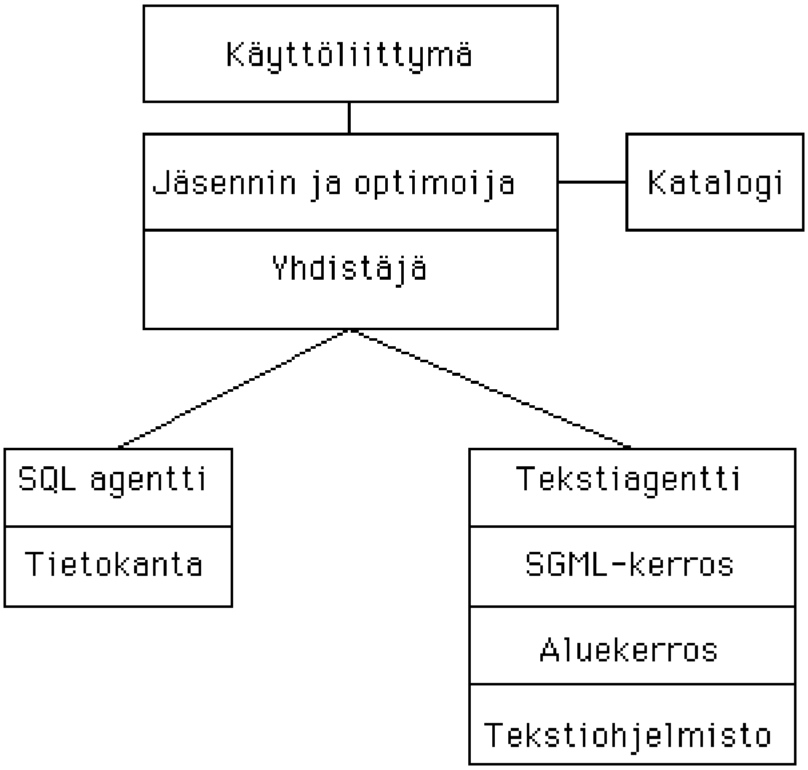
4.2 Tutkimusta

Tutkimustietoa relaatiotietokantojen laajentamisesta käyttämään hyväksi SGML-dokumenttien rakenteita hyväksi on artikkelissa *Text/Relational Database Management Systems: Harmonizing SQL and SGML* [Bla94]. Siinä on esitetty lisäyksiä relaatiotietokantojen standardoituneeseen *Structured Query Language* (SQL) [ISO90] -kieleen. Esityksessä on myös uusi tietotyyppi — TEXT — jolle voidaan määritellä erityinen kielioppi eli käytännössä DTD. Artikkelin mukaisesta relaatiotietokannasta voitaisiin etsiä dokumentteja tarkempien tietojen kuin pelkkien teksti- tai numerovertailujen perusteella. Tästähän oli jo aiemmin puhetta (ks. 4.1).

Edellä mainittu artikkeli on siitä mielenkiintoinen, että siinä pyritään mahdollisimman pienillä muutoksilla saamaan jo käytössä olevat standardit ja ohjelmistot tukemaan myös rakenteisten tekstien tehokasta talletusta ja käyttöä tietokannoissa. Koska tietokannoilla on jo tehokkaat hakumenetelmät, tutkijat ehdottavatkin jo olemassa olevien menetelmien mahdollisimman tehokasta hyväksikäyttöä. Käytännössä tämä tarkoittaa SGML-dokumenttien jäsentämistä ja syntyvien jäsennyspuiden muuntamista relaatioiksi.

Toisaalta voitaisiin ehkä vieläkin helpottaa siirtymistä teksti-/relaatiotietokantaan integroimalla nykyisiä tietokantoja ja tekstinkäsittelyjärjestelmiä. Kuvassa 2 on em. artikkelin kirjoittajien malli tästä ideasta [Bla94]. Käyttöliittymä tarjoaa ohjelmistoille tavan käyttää alla olevia tasoja yksityiskohtiin sekaantumatta. Jäsennin tarvitaan mm. tulkitsemaan käyttöliittymältä tulevien komentojen oikeellisuus. Optimoijaa tarvitaan valittaessa sopivat esitysmuodot komentojen toteuttamista varten tietokanta- ja tekstisysteemeissä. Tässä optimoija käyttää apuna katalogia. Yhdistäjä jakaa komennot tietokanta- ja tekstisysteemille tarpeen mukaan, ja yhdistää niiden palauttamat tiedot. SQL-agentti ja tekstiagentti ovat alla olevasta järjestelmästä riippumattomia, joiden tehtävänä on tarjota standardimuotoinen liittymä yläpuolella olevalle tasolle. Jäljellä olevat tekstipuolen kerrokset muuttavat hakukomennot tekstiohjelmalle sopivaksi.

Myös uudemmat tutkimukset viittaavat siihen, että pienillä SQL-kielen laajennuksilla päästäisiin hyviin tuloksiin SGML-dokumenttien kanssa [Sen96a]. Eräs laajennus Senguptan mallissa on esimerkiksi tapa viitata SGML-dokumentin yksittäisiin elementteihin. SQL ei nykyisellään sovellu erityisen hyvin rakenteisten dokumenttien käsittelyyn "matalan rakennepuunsa" vuoksi: SGML-dokumentti voidaan kyllä hajottaa tauluiksi ja relaatioiksi, mutta se ei ole



Kuva 2: Yhdistetyn tietokannan ja tekstiohjelmiston malli

enää sellainen yhtenäinen kokonaisuus, joksi dokumentti yleensä mielletään. SQL on kuitenkin hyvä lähtökohta laajennukselle, koska se on jo kohtalaisen kauan käytössä ollut standardi. Lisäksi SQL-kieltä on helppo ymmärtää ja käyttää, koska se muistuttaa läheisesti englantia. SQL-kieli on hyvä myös siinä mielessä, että SQL-haut voidaan toteuttaa polynomisessa ajassa ja logaritmisessa tilassa [Sen96b].

Artikkelissa *Standardizing the Querying Process with SGML* Arijit Sengupta esittää, miten laajennetulle SQL-kielelle voidaan kirjoittaa DTD, jolloin tietokantahaut voidaan kirjoittaa SGML:llä [Sen96b]. Tällä on tietysti itsestään selviä hyviä puolia: helppo ymmärtää ja visualisoida, hautkin voidaan tallentaa SQL/SGML-tietokantaan ja siirrettävyys on hyvä. Sengupta vertaa SGML DTD:llä tehtyjä hakuja mm. DSSSL:n *Standard Document Query Language* (SDQL) -kieleen. SDQL sopii hyvin dokumenttien tyylien manipuloimiseen, mutta sen alhainen taso tekee siitä hankalasti käytettävän hakukielen. Esimerkiksi liitosoperaatioita varten joudutaan SDQL:ää käytettäessä ohjelmoimaan LISP-kielellä liitosalgoritmi.

4.3 Multidoc PRO

Vaasalaisen CITEC Engineering Oy:n informaatioteknologian osasto on kehittämässä SGML-selailuohjelmaa Microsoftin Windows-ympäristöön. Ohjelma on nimeltään Multidoc PRO. SGML-osien pohjana on Synexin ViewPort [Syn96] eli sama kuin Panorama PRO:lla [Syn94]. Multidoc PRO oli jo alpha-vaiheessaan kehittyneempi kuin 1.11 versio Panorama PRO:sta (WWW-yhteyttä lukuun ottamatta). Multidoc PRO:ssa on esimerkiksi ensimmäisenä SGML-selailuohjelmana maailmassa mahdollisuus saada useita “näkymiä” dokumenttiin elementtien attribuuttien arvojen perusteella. Tällaisia näkymiä voivat olla esimerkiksi erikieliset dokumentit: käyttäjä voi avata dokumentin ja katsoa toisessa ikkunassa espanjankielistä ja toisessa ikkunassa englanninkielistä dokumenttia.

Multidoc PRO:n tietokantaliittymä toteutetaan Microsoftin *Open Database Connectivity* (ODBC) [Mic92] -standardin mukaan, joka mahdollistaa monien eri valmistajien tietokantojen käytön. Riittää, että tietokannan valmistajalla on tarjota ODBC-ajuri.

Multidoc PRO:n idea tietokantadokumentin muodostamisessa on se, että muodostetaan ensin ns. pseudo-SGML -dokumentti, jossa on vain linkitetty SGML-elementit tietokannan sarakkeisiin. Varsinaisen tietokanta-SGML -dokumentin avautuessa Multidoc PRO suorittaa tietokantahaut, ja täyttää tietokannan tiedoilla “aukot” pseudo-SGML -dokumentissa. Lisäksi pseudo-SGML -dokumentti tarvitsee muodostaa vain kerran. Tietokannassa tehdyt muutokset saadaan näkyviin Multidoc PRO:ssa yksinkertaisesti valitsemalla dokumentin “virkistys”.

4.4 Astoria

Astoria on oliopohjainen SGML-tietokanta. Mahdollisesti oliopohjaiset tietokannat yleensäkin voisivat olla paremmin sopivia rakenteisen tekstin kanssa toimimiseen, ovathan SGML-elementit selkeitä kokonaisuuksia.

Astorian erityispiirteitä on mm. se, että siltä voi pyytää SGML-dokumentin osan vaikkapa editoimista varten. Harvinaista tässä on se, että Astoria osaa liittää dokumentin mukaan alkuperäisestä DTD:stä kootun osasen, joka riittää kuvailemaan osadokumentin. Tällä on tietysti merkitystä tehokkuuden kanssa silloin, kun editointeja tehdään paljon tai kun DTD on suuri.

Astorian Systems Development Kit (SDK) ja API ovat hyvin monipuolisia. Itse asiassa niillä voi aivan hyvin kehittää itse paremmat ohjelmistot tietojen käsittelyyn kuin mukana seuraavat valmisohjelmistot antavat myöten.

5 Muut muunnokset

Your quotation here. *Bjarne Stroustrup*

Perinteisesti SGML-dokumentteja on jouduttu muuntamaan muihin muotoihin julkaisua varten. Yleisimpiä näistä muutoksista lienevät LaTeX-muunnokset ja tietenkin nykyään HTML-muunnokset. Toisaalta myös ei-rakenteista ja ei-SGML -muotoista dokumentaatiota muutetaan SGML:ksi.

5.1 SGML:stä julkaisuksi

Koska SGML-dokumentit ovat pelkkää tekstiä rakennemerkintöjen lisäksi, niitä ei kannata sellaisenaan tulostaa tai laittaa näytille minnekään. Julkaisua varten onkin jouduttu muuttamaan SGML-dokumentti ensin sellaiseen muotoon, että se voidaan tulostaa tai esittää näyttöpäätteellä. Jo edellä mainitut Linuxdoc-SGML ja Sgmlspl pystyvät tähän, Linuxdoc-SGML suoraan formaateiksi LaTeX, groff ja puhdas ASCII-teksti.

Nämä muunnokset asettavat tietysti erilaisia vaatimuksia muunnosten suhteen kuin puhtaat SGML-muunnokset. Koska SGML:ssä ei oteta tekstin asetteluun mitään kantaa, joutuu käyttäjä itse määrittelemään, kuinka hän haluaa tietyt elementit esitettävän. Toisaalta, jos on käytössä jokin tunnettu DTD ja tiedetään, mihin formaattiin se halutaan muuttaa, voidaan tehdä erikoiskäyttöinen ohjelma hoitamaan tämä automaattisesti.

Nykyään ollaan myös siirtymässä järjestelmiin, joissa SGML-dokumenttien tekstin asettelu voidaan kuvata mukana seuraavalla tyyli-DTD:llä tai tyylidokumentilla. Vakiintunutta käytäntöä näitä varten ei vielä ole, vaan kukin valmistaja esittää asiat omalla tavallaan. Siisti toteutus on esimerkiksi Synex ViewPort -tuotteessa [Syn96].

Standardien kehitys on tässä jäänyt ehkä hieman jälkeen, mutta *Document Style Semantics and Specification Language* (DSSSL) [ISO95] antanee osviittaa siitä, mihin ollaan menossa. DSSSL-standardi määrittelee kyllä muutakin kuin dokumenttien tyylejä: sen Turing-täydellinen kieli esimerkiksi mahdollistaa periaatteessa minkä tahansa DTD:n mukaisen osadokumentin luomisen olemassa olevasta SGML-dokumentista. Eli samasta SGML- dokumentaatiosta voidaan DSSSL:n avulla tehdä kaksi “näkymää”, toinen esimerkiksi tehtaan johtajalle ja toinen työntekijälle. Ongelmiakin tässä on. On nimittäin huomattu, että ei ole mahdollista laskennallisesti päätellä, vastaako jokin mielivaltainen muunnos jotakin mielivaltaista DTD:tä [Ray96].

5.2 Erilaisista formaateista SGML:ksi

Verkosta löytyvät ohjelmat, jotka kuuluvat kategoriaan "muunna jotakin SGML:ksi", muuntavat dokumentteja yleensä HTML:ksi. Tämän muunnoksen jälkeen oltaisiinkin taas puhtaan SGML-muunnoksen alueella. Tässä kuitenkin usein menetään informaatiota — sitä enemmän mitä useampia muutoksia välillä tehdään. Näitä ohjelmia on kuitenkin eniten tarjolla.

Verkosta löytyviin muunnosohjelmiin on syytä suhtautua kriittisesti. Xavier Leroyn sanoin:

> Many are quick hacks that have been put together in a hurry and made public after very little testing. "It works for me; have fun!".

Tekstin muuntamisohjelmia HTML:ksi löytyy useita. Niissä täytyy kuitenkin tehdä jonkinlaisia olettamuksia tekstistä, esimerkiksi tyhjä rivi tarkoittaa uutta kappaletta, yksinäinen rivi tarkoittaa otsikkoa, tähdellä alkava rivi listan alkiota jne. Tekstin muuttamisessa ollaan selvästi tilanteessa, jossa alkuperäisessä formaatissa on vähemmän informaatiota itse tekstistä kuin muunnoksen kohdeformaatissa.

LaTeXin muuntaminen HTML:ksi on tavallaan helpompaa kuin pelkän tekstin. Toki LaTeXin erikoispiirteet vaikeuttavat asiaa, mutta silti LaTeXissa on myös rakenneinformaatiota, esimerkiksi otsikoiden tasot. LaTeXin matemaattiset kaavat ovat olleet suurin haitta hyvilläkin muunnosohjelmilla, sillä yleensä ne on vain muunnettu kuviksi HTML:n heikon matemaattisten lausekkeiden tukemisen vuoksi. Tämä aiheuttaa ensinnäkin heikomman esteettisen tuloksen, ja lisäksi kuvat vievät enemmän tilaa ja niiden siirtämiseen menee enemmän aikaa kuin pelkkään tekstiin. Xavier Leroyn kirjoitus on opettavainen tarina LaTeX muunnosten hankaluudesta [Ler].

Erilaisia Help-formaatteja varten on myös olemassa paljon muunnosohjelmia. Sekä Windowsin Help että UNIXin man-sivut on mahdollista saada HTML:ksi. Näillä on ehkä enemmänkin merkitystä intranettien yleistyessä, kukapa haluaisi kalliin ohjelmansa osia jakaa ilmaiseksi ulkopuolisille.

Jopa lähes pelkästään dokumentin visuaalisesti kuvailevista formaateista on mahdollista saada enemmän tai vähemmän kelvollista-HTML tekstiä. PostScript, *Rich Text Format* (RTF) ja groff ovat vain esimerkkejä monista mahdollisuuksista.

6 Yhteenveto

> Humankind periodically goes through a speedup of its affairs, thereby experiencing the race between the renewable vitality of

> the living and the beckoning vitiation of decadence. In this periodic race, any pause becomes luxury. Only then can one reflect that all is permitted; all is possible. *Frank Herbert, Children of Dune*

SGML:ksi muuntamisen merkitys kasvanee muutaman vuoden ajan voimakkaasti. Toisaalta julkaisutarve WWW:lle aiheuttaa ainakin toistaiseksi suurta muunnostarvetta myös HTML:ksi, mutta toivottavasti tässäkin kehitys menee eteenpäin ja WWW:llä voidaan näyttää suoraan oikeita SGML-dokumentteja. Suuntaa näyttävä ohjelma tässä oli SoftQuadin Panorama PRO, ensimmäinen WWW:tä tukeva SGML-selainohjelma [Syn94].

Toisaalta viime aikoina suurta kiinnostusta on herättänyt tekeillään oleva standardi *Extended Markup Language* (XML) [Con97]. Sen ennustetaan korvaavan HTML World Wide Webin dokumenttien kuvailukielenä. XML on yksinkertaistettu SGML-standardi ja SGML:n kanssa yhteensopiva siten, että XML-dokumentit ovat täysin SGML:n mukaisia. Siis SGML-ohjelmistoilla voidaan käsitellä XML-dokumentteja, mutta ei päinvastoin. XML:stä on siten helppo siirtyä puhtaaseen SGML:ään.

Rakenteisten dokumenttien määrä kasvaa nopeammin nyt, kun niiden edut on ymmärretty ja niiden käsittelyä varten alkaa olla ohjelmistoja. Myös kaupallisten ohjelmistojen osuus lienee jo ohittanut vapaassa levityksessä olevien ohjelmien lukumäärän.

Omalta osaltaan myös WWW on varmasti vauhdittanut SGML:n yleistymisessä. Tämä luonnollisesti vähentää muunnoksien tarvetta, mutta kaikkea dokumentointia saadaan tuskin milloinkaan rakenteisten standardien piiriin. SGML:ää käytettäessä ongelmaksi jäävät myös erilaiset DTD:t, vaikka monia kohtuullisen yleiskäyttöisiä DTD-standardejakin on jo syntynyt. Näin on aina tarvetta myös muunnoksille.

Lähteet

[Akp92] L. Akpotsui & V. Quinn. *Type Transformations in Structured Editing Systems.* Electronic Publishing — Proceedings of Electronic Publishing. Cambridge University Press, 1992.

[Arb95] ArborText, Inc. *Getting Started with SGML — a Guide to the Standard Generalized Markup Language and Its Role in Information Management*, 1995. An ArborText White Paper.

[Bla94] G. E. Blake & M. P. Consens & P. Kilpeläinen & P. A. Larson & T. Snider & F. W. Tompa. *Text/Relational Database Management Systems: Harmonizing SQL and SGML.* Proc. Applications of Databases (ADB–94). Vadstena, June 1994.

[Col92] H. Brown & F. Cole. *Editing Structured Documents — Problems and Solutions*, volume 5 of *Electronic Publishing.* John Wiley & Sons, Ltd., December 1992.

[Con97] D. Connolly & J. Bosak. *Generic SGML over the Web.* 1997. http://www.w3.org/pub/WWW/MarkUp/SGML/Activity

[Fre] E. D. Freese. *The Transformation of SGML Documents for Presentation on the World Wide Web.* http://141.142.3.76/SDG/IT94/Proceedings/Autools/freese/freese.html.

[Her94] E. van Herwijnen. *Practical SGML.* Kluwer Academic Publishers, second edition, 1994.

[ISO86] ISO. *Information Processing — Text and Office Systems — Standard Generalized Markup Language*, 1986. ISO 8879.

[ISO89] ISO. *Information Processing — Text and Office Systems — Office Document Architecture*, 1989. ISO 8613.

[ISO90] ISO. *Information Technology — Database Language SQL 2 Draft Report*, 1990. ISO Committee ISO/IEC JTC 1/SC 21.

[ISO92] ISO. *Information Technology — Hypermedia/Time-based Structuring Language*, 1992. ISO/IEC 10744.

[ISO95] ISO. *Information Technology — Text and Office Systems — Document Style Semantics and Specification Language*, 1995. Working Draft, ISO/IEC DIS 10179.2.

[Kui93] E. Kuikka & M. Penttonen. *Transformation of Structured Documents With the Use of Grammar*, volume 6 of *Electronic Publishing*. John Wiley & Sons, Ltd., December 1993.

[Lam94] L. Lamport. *LaTeX: A Document Preparation System*. Addison–Wesley Publishing Company, Inc., 1994.

[Ler] X. Leroy. *Lessons Learned from the Translation of Documentation from LaTeX to HTML*. http://pauillac.inria.fr/~xleroy/w4g.html.

[Mic92] Microsoft Corporation. *Microsoft ODBC Application Programmer's Guide*. Microsoft Corporation, 1992.

[Kun95] R. Kunnari & V. Lyytikäinen & T. Pietinen. *Digitaalinen julkaiseminen — sovellusalueena yliopiston julkaisutuotanto*, 1995. Tietojärjestelmätieteen pro gradu -tutkielma.

[Ray96] D. Raymond. & F. Tompa & D. Wood. *From Data Representation to Data Model: Meta-Semantic Issues in the Evolution of SGML*, volume 18 of *Computer Standards & Interfaces*. 1996.

[Sen96a] A. Sengupta. *Demand More from Your SGML Database! Bringing SQL under the SGML Limelight*, in <TAG> *The SGML Newsletter*. April, 1996.

[Sen96b] A. Sengupta. *Standardizing the Querying Process with SGML: The SQL DTD*, in *SGML '96 Conference Proceedings in Boston*. Alexandria, WA. GCA, 1996.

[Syn94] Synex Information AB. *Panorama PRO Guide*, 1994.

[Syn96] Synex Information AB. *Synex ViewPort Developers Manual*, 1996.

[TEI] Text Encoding Initiative. *A Gentle Introduction to SGML*. http://etext.virginia.edu/bin/tei-tocs?div=DIV1&id=SG.

[Toi95] H. Toivonen & M. Hirvonen. *Automatic Transformation of Structured Documents*, 1995. http://www.jyu.fi/~hjtoi/auto.html.

A Ohjelmistoja

Alla on lueteltu, mistä saa lisätietoja tekstissä mainituista ohjelmista. Kaupallisistakin ohjelmista saa usein ladattua näytekappaleet.

Ohjelma	URL
Astoria	http://www.chrystal.com/product.htm
Fred-projekti	http://www.oclc.org/fred/
LaTeX2HTML	http://cbl.leeds.ac.uk/nikos/tex2html/doc/latex2html/latex2html.html
Linuxdoc-SGML	http://www.linux.locus.halcyon.com/Linuxdoc-SGML.htmls
Multidoc PRO	http://www.citec.fi/mdp/index.html
Panorama PRO	http://www.synex.se/panorama.html
Sgmlspl	http://www.uottawa.ca/~dmeggins/sgmlspl/sgmlspl.html

B Dokumenttityyppimääritykset

DTD	URL
QWERTZ	ftp://ftp.ifi.uio.no/pub/SGML
HTML	http://www.w3.org/pub/WWW/MarkUp/html-spec/html-spec_toc.html